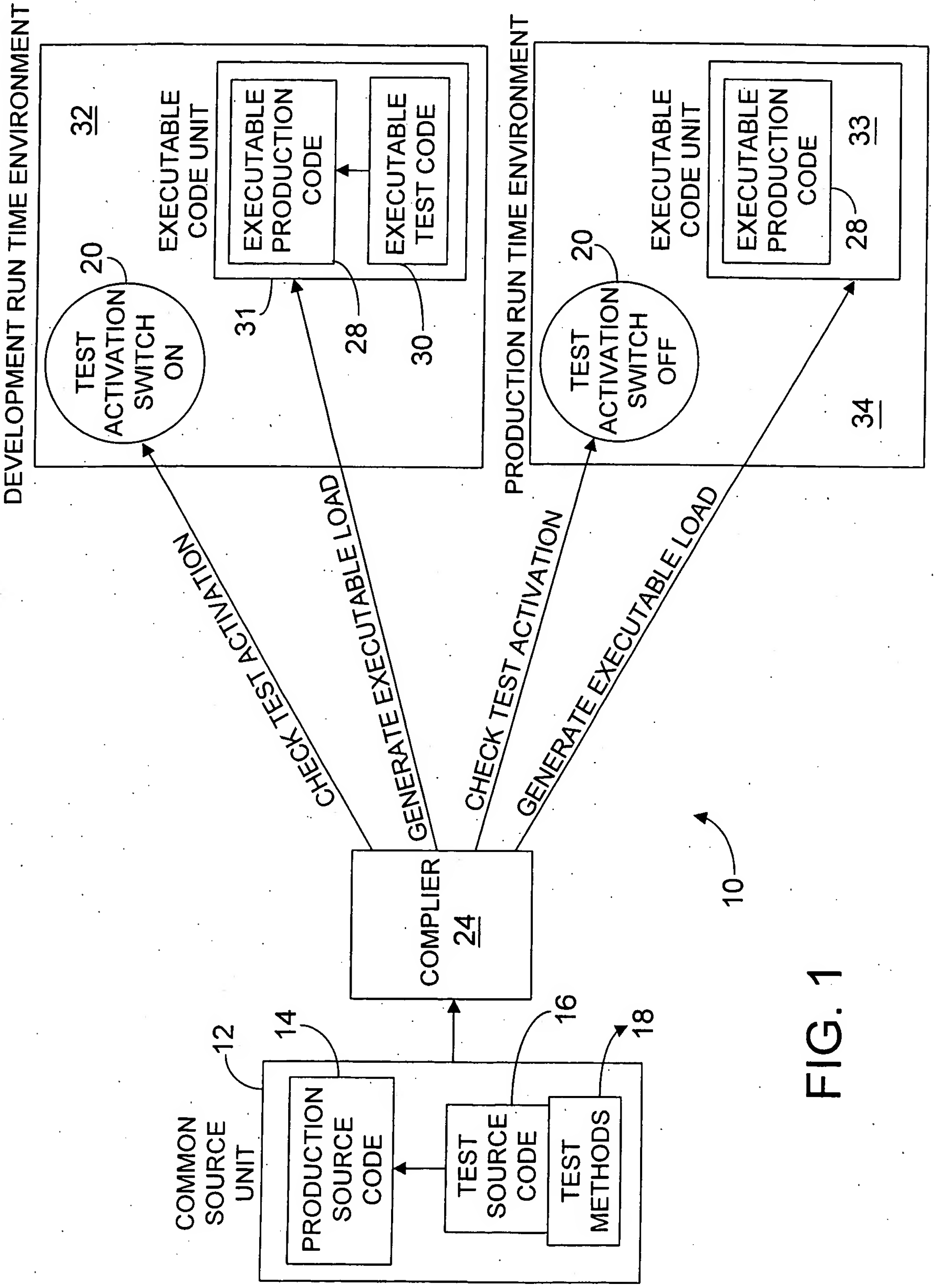
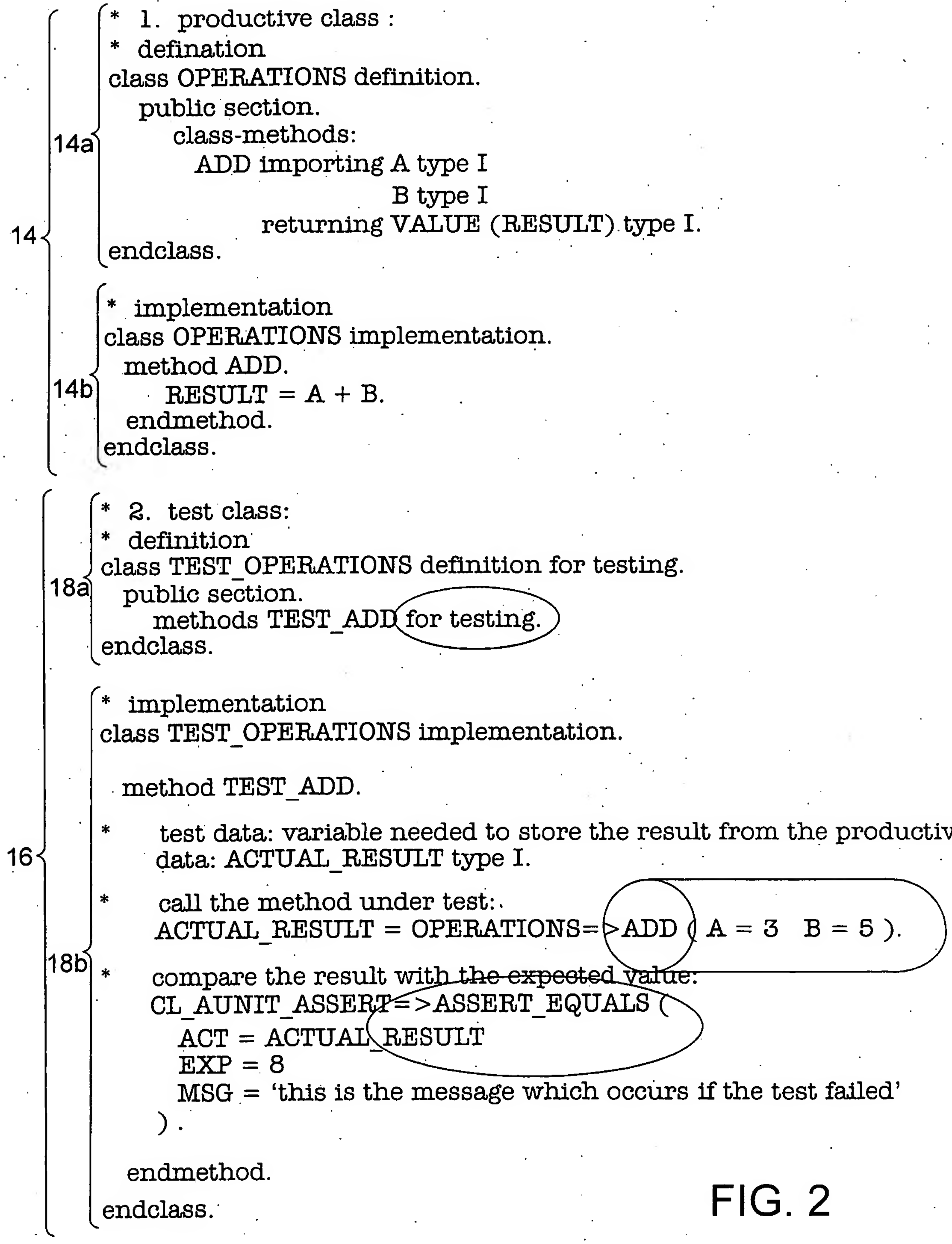

DEVELOPMENT RUN TIME ENVIRONMENT
32
TEST ACTIVATION SWITCH ON
20
EXECUTABLE CODE UNIT
EXECUTABLE PRODUCTION CODE
EXECUTABLE TEST CODE
31
28
30
PRODUCTION RUN TIME ENVIRONMENT
TEST ACTIVATION SWITCH OFF
20
EXECUTABLE CODE UNIT
EXECUTABLE PRODUCTION CODE
33
28
34
CHECK TEST ACTIVATION
GENERATE EXECUTABLE LOAD
CHECK TEST ACTIVATION
GENERATE EXECUTABLE LOAD
COMPLIER
24
10
COMMON SOURCE UNIT
12
14
PRODUCTION SOURCE CODE
TEST SOURCE CODE
16
TEST METHODS
18

FIG. 1



```
* 1. productive class :
* defination
class OPERATIONS definition.
   public section.
      class-methods:
        ADD importing A type I
                      B type I
              returning VALUE (RESULT) type I.
endclass.

* implementation
class OPERATIONS implementation.
  method ADD.
     RESULT = A + B.
  endmethod.
endclass.

* 2. test class:
* definition
class TEST_OPERATIONS definition for testing.
  public section.
     methods TEST_ADD for testing.
endclass.

* implementation
class TEST_OPERATIONS implementation.

  method TEST_ADD.

*     test data: variable needed to store the result from the productive method
      data: ACTUAL_RESULT type I.

*     call the method under test:
      ACTUAL_RESULT = OPERATIONS=>ADD ( A = 3   B = 5 ).

*     compare the result with the expected value:
      CL_AUNIT_ASSERT=>ASSERT_EQUALS (
        ACT = ACTUAL_RESULT
        EXP = 8
        MSG = 'this is the message which occurs if the test failed'
      ) .

  endmethod.
endclass.
```

FIG. 2

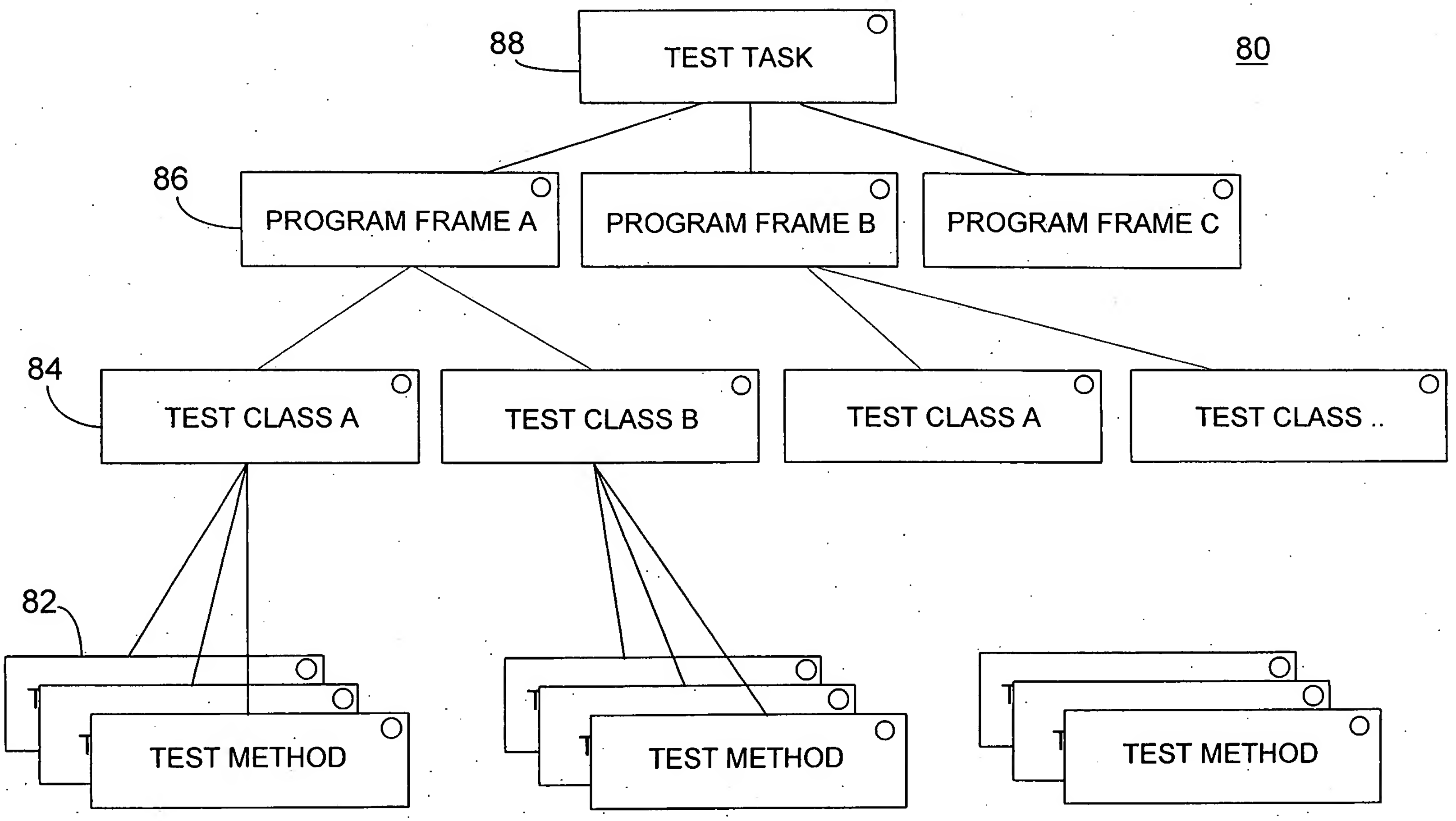

88
TEST TASK
80
86
PROGRAM FRAME A
PROGRAM FRAME B
PROGRAM FRAME C
84
TEST CLASS A
TEST CLASS B
TEST CLASS A
TEST CLASS ..
82
TEST METHOD
TEST METHOD
TEST METHOD

FIG. 3

$$\overbrace{\text{ASSERT\_EQUALS}}^{56}\ (\ \text{ACT} = \text{ACTUAL RESULT}$$

$$\begin{array}{l} \text{EXP} = \text{EXPECTED\_RESULT} \\ \text{MSG} = \text{'this test has failed'} \\ \underbrace{\text{QUIT}}_{57} = \underbrace{\text{QUIT\_VALUE}}_{58}\ ). \end{array}$$

Where QUIT_VALUE defines at which level the test flow should be interupted:

- NO: continue the current test method.
- METHOD: interrupt the current test method.
- CLASS: interrupt the test class execution.
- PROGRAM: abandon all test class executions of the currently tested program frame.

FIG. 4

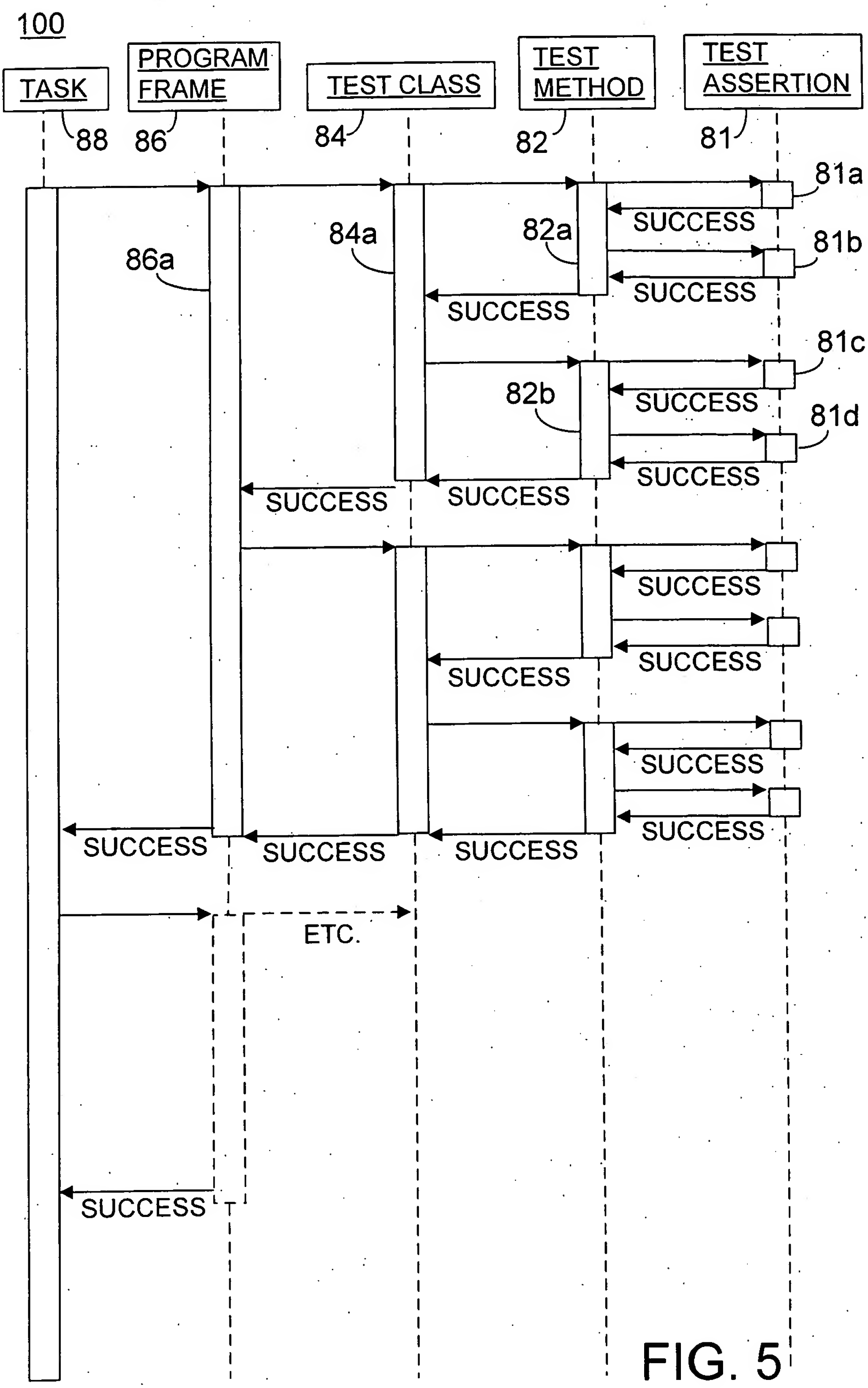

100
TASK
PROGRAM FRAME
TEST CLASS
TEST METHOD
TEST ASSERTION
88
86
84
82
81
81a
SUCCESS
81b
84a
82a
86a
SUCCESS
SUCCESS
81c
82b
SUCCESS
81d
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
SUCCESS
ETC.
SUCCESS

FIG. 5

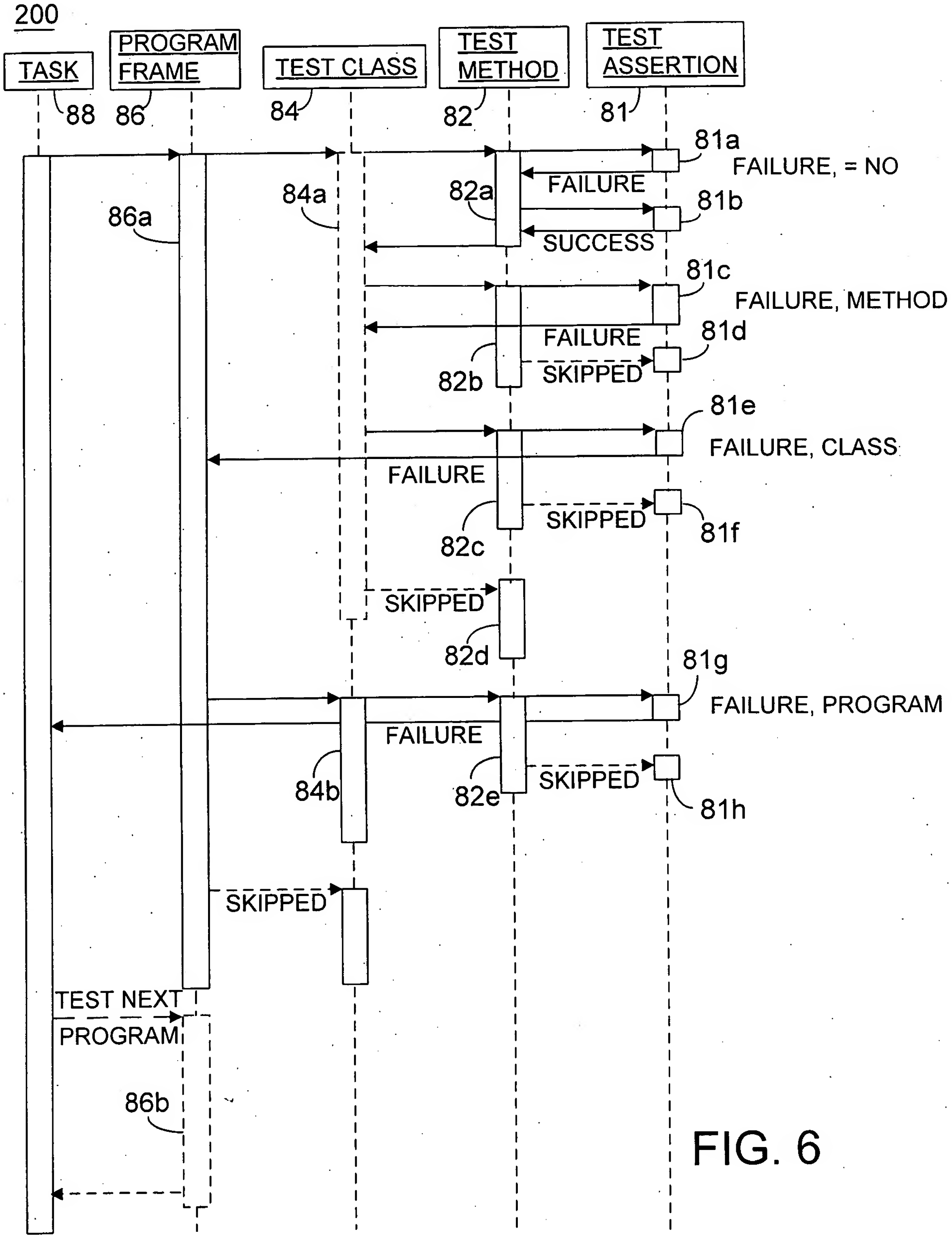

200
TASK
PROGRAM FRAME
TEST CLASS
TEST METHOD
TEST ASSERTION
88
86
84
82
81
81a
FAILURE, = NO
FAILURE
84a
82a
81b
86a
SUCCESS
81c
FAILURE, METHOD
FAILURE
81d
82b
SKIPPED
81e
FAILURE, CLASS
FAILURE
SKIPPED
82c
81f
SKIPPED
82d
81g
FAILURE, PROGRAM
FAILURE
84b
82e
SKIPPED
81h
SKIPPED
TEST NEXT
PROGRAM
86b

FIG. 6